Assignment 2
CSL7590: Deep Learning
AY 2023-24, Semester – II
Due on: 15-02-2024

M.M: 200

General Instructions:

1. Clearly mention the assumptions you have made, if any.
2. Clearly report any resources you have used while attempting the assignment.
3. Any submission received in another format or after the deadline will not be evaluated.
4. Make sure to add references to the resources that you have used while attempting the assignment.
5. Plagiarism of any kind will not be tolerated and will result in zero marks.This is applicable to both code and report.
6. Select your dataset correctly. If found otherwise, your assignment will not be evaluated.
7. Late submission penalty will be calculated as a 20% deduction per day.

Submission Guidelines:

1. Prepare a Python code file for the task and name it as **YourRollNo.py.** There should be **one and only one .py file**. No need to prepare a separate .py file per subtask. **The .py files must not be named like <roll no>_task1(1).py**

2.Submit a single report depicting methods, results, and observations. There is no need to add theory behind the concepts. Preparing a report is mandatory; failing it will lead to non-evaluation of the assignment.

3. Name your report as **YourRollNo.pdf.** Also, **provide your colab file link in the report.** Make sure that the file is sharable.
4. There is **no need to make a zip file.** Just upload both the codes and a report directly on the google-classroom, that is, submission will contain {YourRollNo.py and YourRollNo.pdf}. **Do not upload files in any other format.**
5. Do not download the .ipynb file, rename it as .py, and upload it. .ipynb files are not exactly in a readable form, so uploading it will only result in you receiving 0 marks for the same. You have an option to download a .py file in google colab. Use it to get the .py format.
6. Do not copy-paste code snippets or screenshots, etc. in the report. The report should look like a technical document, containing plots, tables, etc. whenever necessary.
7. **Adhere to the instructions given, failing them may result in a penalty.**

Objective:

In this assignment, you are required to implement a transformer network in Python using the Pytorch framework. By the end of this assignment, you should have a working transformer network that can be trained on a simple audio dataset for multi-class classification.

Dataset:

Get the dataset from here. The dataset consists of 400 environmental audio recordings categorized into 10 different classes.

Pre-processing:

You are being provided a sample dataloading and preprocessing code which serves as a reference. Study the code, understand the guidelines and feel free to write your own custom dataloader within those boundaries. The code is available here, which you may use as it is, however, you are encouraged to do it on your own.

Network Architecture: [25 + 35 = 60 marks]

Architecture 1

Use 1D-convolution for feature extraction. The base network should be at least three layers deep. On top of it implement a fully-connected layer for multi-class classification. You are free to use any number of layers, strides, kernel size, number of filters, activation functions, pooling, and other free parameters. **Use Pytorch only.**

Architecture 2

- Use 1D-convolution for feature extraction. Use the same base network as above. On top of the base network, implement a transformer encoder network (from scratch) with a multi-head self-attention mechanism. Make sure to add the <cls> token. On the top of the transformer, an MLP head should be there for classification. Your model should have at least two attention blocks for number of heads = 1,2,4.
- Prepare a detailed analysis, which model achieves best accuracy and why?
- The self-attention is to be implemented by scratch to solve a 10-class classification problem using Pytorch. **Use Pytorch/Pytorch Lightning.**

Tasks: [70 marks]

1. Train for 100 epochs. Plot accuracy and loss per epoch on Weight and Biases (WandB) platform. Also, mention plots in the report.
2. Perform k-fold validation, for k=4.
3. Prepare an Accuracy, Confusion matrix, F1-scores, and AUC-ROC curve for the test set for all the combinations of the network. You may use an in-built function for this purpose.
4. Report total trainable and non-trainable parameters.
5. Perform hyper-parameter tuning and report the best hyper-parameter set.

Perform all tasks for both architectures with both test-split configurations and k-fold validation strategies.

Report: **[40 marks]**

Prepare a detailed report containing all the results, graphs, plots, methodology, and observations. The report should contain a comparative table containing results for all the model configurations.

Demonstration: **[30 marks]**

Evaluation of the assignment will accompany a demonstration of the working model.

Note: Do not use in-built functions unless mentioned.